Structure of Operating system

An Operating System (OS) structure defines how its components are organized and interact to manage hardware and software resources efficiently.

Components of an OS : Kernel , System Calls , Memory Management, Process Management , File System, Device Drivers , User Interface

Types of OS Structures

- Simple Structure
- Monolithic Structure
- Layered Structure
- Micro-Kernel Structure
- Hybrid-Kernel Structure
- Modular Structure
- Virtual Machines

Structure of Operating system

1. Simple Structure : A Simple Structure OS lacks a clear modular design, making its components tightly integrated. It is small, fast, and easy to develop, but less secure and unstable.
Example: MS-DOS

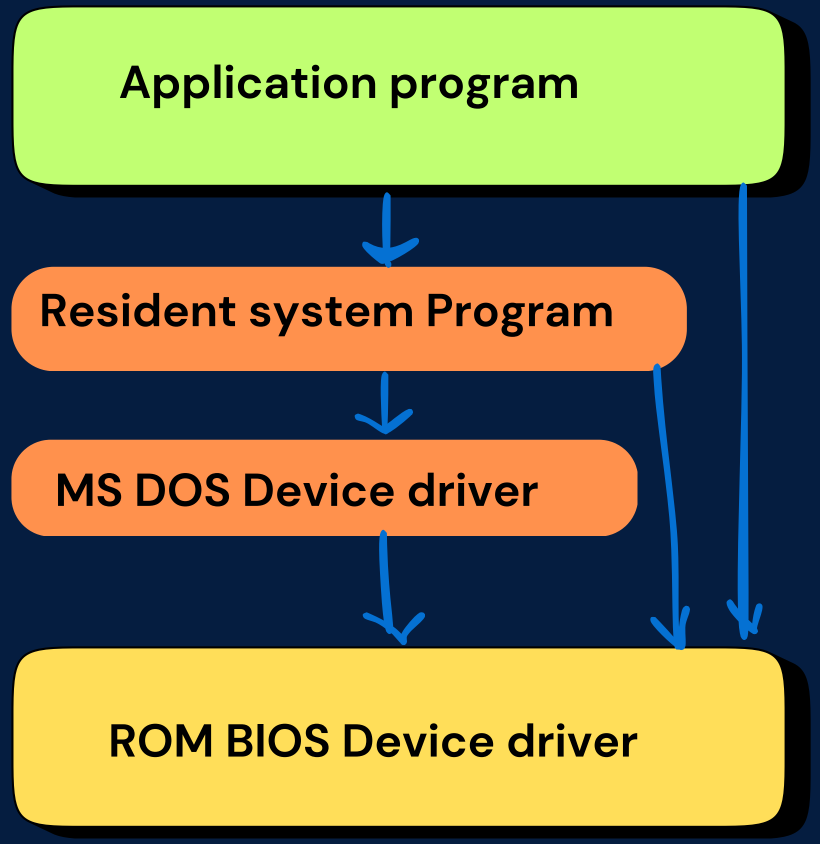
- Allows applications to directly access hardware.
- A single program failure can crash the entire system.

Advantages

- Fast performance due to fewer layers.
- Easy to develop for kernel programmers.

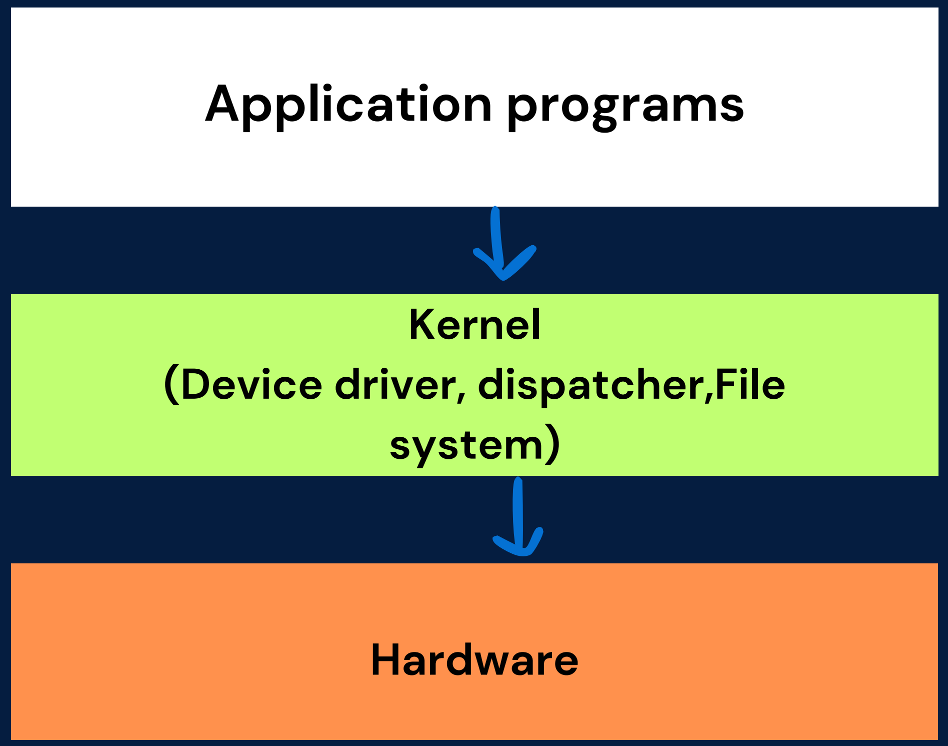
Disadvantages

- No clear structure, making it complex to manage.
- Lacks security as processes can directly access system resources.



Structure of Operating system

2. Monolithic Structure : A Monolithic Structure OS is a single, large program where all essential services (process management, memory management, file systems, and device drivers) run together in kernel mode.
Example: Linux, UNIX



The **Kernel** is the central component of an OS that ensures communication between the hardware and software. It provides fundamental services like process management, memory management, and device communication, ensuring the system runs efficiently and securely.

Advantages

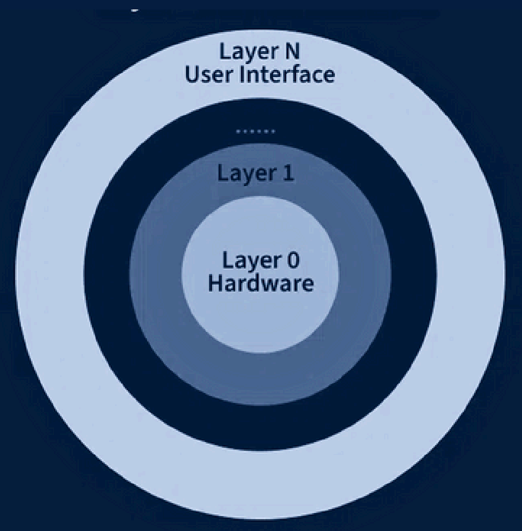
- Fast performance due to direct communication between components.
- Easier to build since everything is in one block.

Disadvantages

- Difficult to maintain as a small error can crash the entire system.
- Security risks due to lack of strict isolation between components.

Structure of Operating system

3. Layered Structure: The OS is divided into multiple layers, each providing services to the layer above it. You can think Layered Architecture like A cake with multiple layers, each performing a specific role.
Eg : Windows NT



Advantages:

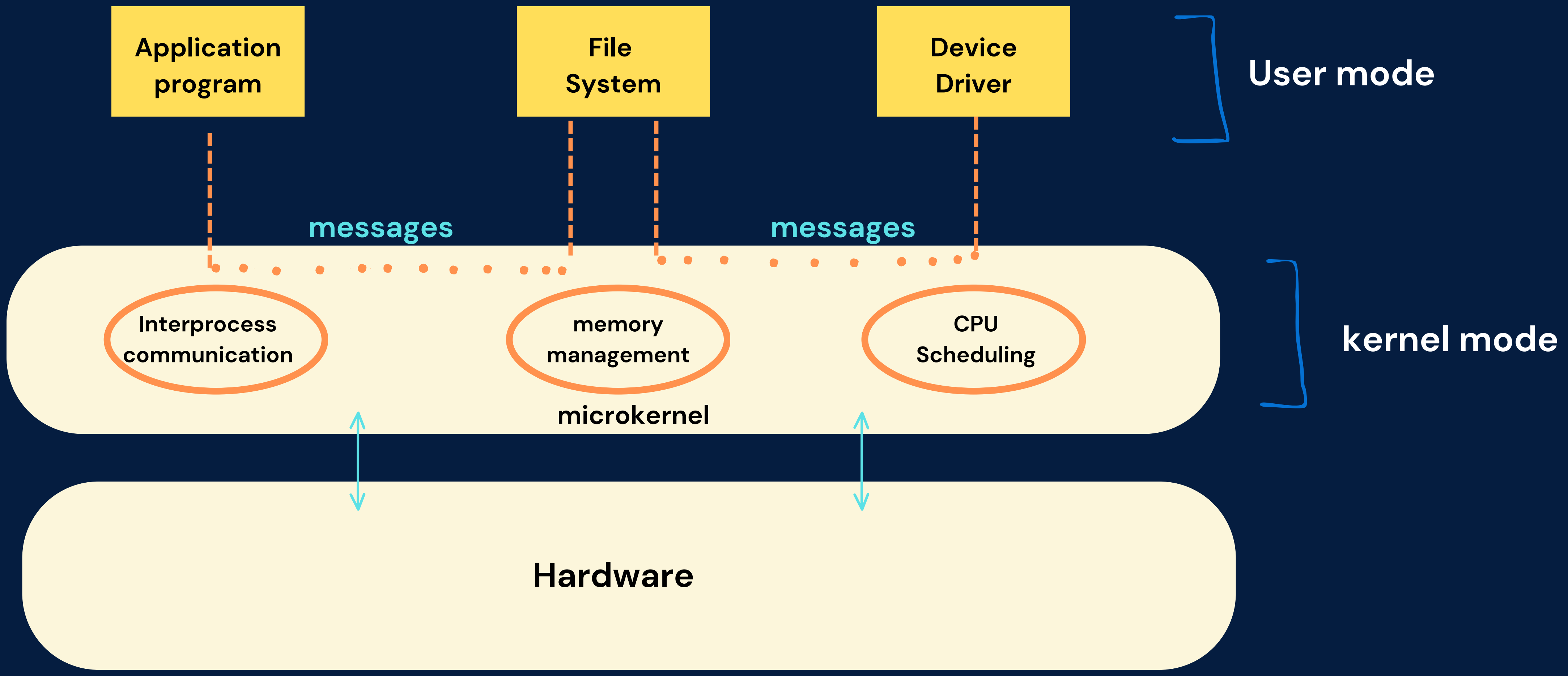
- Easier debugging and modification.
- More secure since lower layers are protected from higher layers.

Disadvantages:

- Performance overhead due to inter-layer communication.
- Strict layering can cause inefficiencies.

Structure of Operating system

4. Microkernel Structure: Only essential OS functions (e.g., process and memory management) run in the kernel, while other services run in user space.. You can think Microkernel Architecture like A company where only critical tasks are handled by the core team, while other tasks are outsourced.
Eg : macOS, QNX, Minix



Structure of Operating system

4. Microkernel Structure

Advantages:

- More stable (a failure in one module won’t crash the whole system).
- Easier to extend and modify.

Disadvantages:

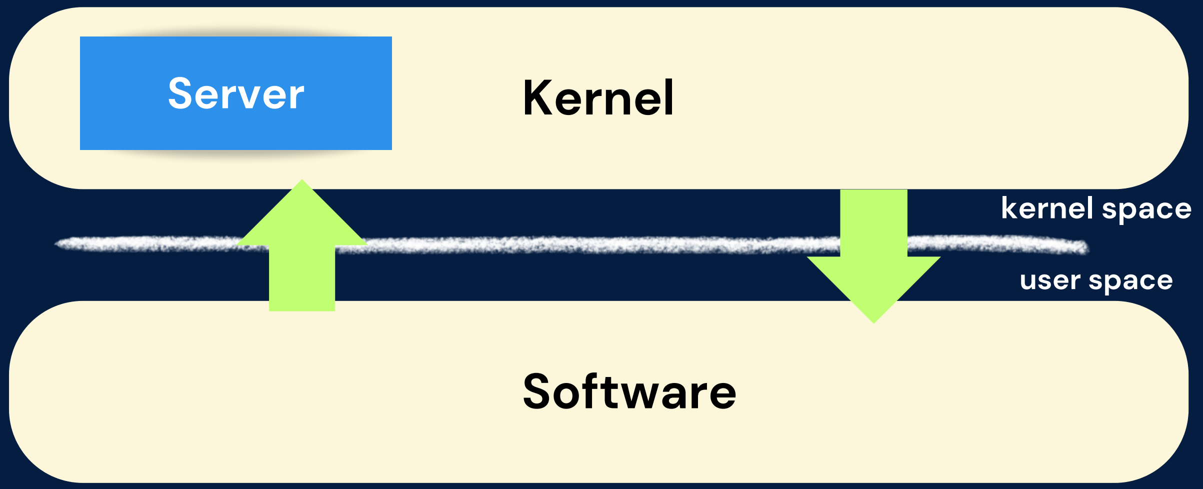
- Slower due to frequent communication between kernel and user space.
- More complex implementation.

Structure of Operating system

- **Microkernel** = Security & modularity first, but with performance trade-offs.
- **Hybrid Kernel** = A middle ground, improving speed while retaining some modularity.

5. Hybrid-Kernel Structure: A Hybrid-Kernel Structure is an operating system architecture that combines features of both monolithic kernels and microkernels, aiming to achieve a balance between performance and modularity.

Hybrid kernels follows the microkernel architecture but includes some traditionally user-space services (like device drivers, file systems, or network stacks) inside the kernel space for better performance. Hybrid kernels aim to reduce inter-process communication (IPC) overhead, which is a major drawback of microkernels.


Server
Kernel
kernel space
user space
Software